Assisted Hardware Selection for Industrial Collaborative Robots

Schou, Casper; Hansson, Michael; Madsen, Ole

Published in:
Procedia Manufacturing

DOI (link to publication from Publisher):
10.1016/j.promfg.2017.07.222

Publication date:
2017

Document Version
Publisher's PDF, also known as Version of record

Available online at www.sciencedirect.com

ScienceDirect

Procedia Manufacturing 11 (2017) 174 – 184

Procedia MANUFACTURING

27th International Conference on Flexible Automation and Intelligent Manufacturing, FAIM2017, 27-30 June 2017, Modena, Italy

Assisted Hardware Selection for Industrial Collaborative Robots

Casper Schou[a,*] , Michael Hansson[a], Ole Madsen[a]

[a]*Department of Mechanical and Manufacturing Engineering, Aalborg University, 9220 Aalborg, Denmark*

Abstract

This paper presents a configuration framework for assisting shop floor operators in selecting a suitable hardware configuration from commercially available components. The primary focus of this work is the modeling of process, product, and equipment knowledge, and the design of a configurator tool implementing this knowledge. The configurator takes process and product information as input and derives a list of suitable components for the operator to choose from. The approach is verified through a preliminary study indicating the feasibility of the approach.

Peer-review under responsibility of the scientific committee of the 27th International Conference on Flexible Automation and Intelligent Manufacturing

Keywords: Industrial robots; collaborative robots; hardware reconfiguration; configurator; ontology

1. Introduction

Traditional, dedicated, and fully automated manufacturing systems require large batch sizes to be economically feasible. They are typically not well-suited for small and medium enterprises (SMEs) and companies with high mix, low volume production. Consequently, there is a strong need for automated manufacturing equipment suitable for smaller batch sizes with frequent changeovers [1,2]. Such equipment need to be responsive and agile with the ability to be efficiently reconfigured and reused [3].

This is not least relevant in the field of robotics. In traditional manufacturing systems, the robot is often fixed to a dedicated, fenced-in workstation doing a single repetitive task. Contrary to traditional industrial robots, collaborative

* Corresponding author. Tel.: +45-20868884.
E-mail address: cs@make.aau.dk

Peer-review under responsibility of the scientific committee of the 27th International Conference on Flexible Automation and Intelligent Manufacturing
doi:10.1016/j.promfg.2017.07.222

robots are intended to operate in the more dynamic production environment of the human operators with greater variety, diverse tasks, and more frequent changeovers.

There are two main objectives in transitioning a robot to a new task; 1) configuring the hardware of the robot, and 2) programming the robot to the new task. In the industry, the variation in tasks is often of such magnitude that they cannot be solved by a single hardware configuration [1]. Thus, the need for hardware reconfiguration is inevitable. This reconfiguration should not be an engineering task, but should be handled by the production staff working alongside the robot. However, this requires new approaches to hardware configuration and operating the collaborative robot as compared to a traditional industrial robot [3].

According to [4], the objective of hardware configuration consists of two sub objectives; 1) selecting a set of suitable modules for the given task, and 2) adding, removing and exchanging hardware modules to obtain the selected configuration. In this paper, we focus on the first of these objectives. Selecting suitable modules for a given task is not trivial as it requires knowledge on both the process, the product to be manufactured (and its components), and the equipment to be reconfigured [5]. Although shop floor operators often possess detailed knowledge on the process and the product, they seldom possess detailed knowledge on the robot equipment domain. Our ongoing research consider a configuration framework assisting the user in selecting a feasible set of modules for a given task. In this paper, we focus on the task of modeling knowledge to support the configuration and how this knowledge can be used in the design of a configurator.

2. Related Work

A key aspect towards a systematic design of reconfigurable robotic systems is that knowledge regarding the capabilities and structure of the system can be captured and utilized to support the process of making design decisions. In recent decades, ontologies have been frequently used as a knowledge modeling method [6]. A recent review of research focusing on ontologies for manufacturing purposes is presented in [6]. With offset in the reviewed literature, [6] classify current research on the topic into the following three categories: 1) development of specific applications, 2) development of domain ontologies, and 3) proposal of core ontologies. Category one represents research in which ontologies are used as a supporting technology. The second category contains research within the knowledge modeling itself, proposing ontologies with knowledge for a particular domain. The third category contains research proposing general, core ontologies applicable to a wide range of domains. In conclusion of the review, [6] emphasizes several shortcomings of the OWL format when it comes to modeling complex products or reasoning over procedural knowledge. Despite these shortcomings, [6] notes that since the OWL format has become a commonly used format in both research and practical applications of manufacturing knowledge it is desirable to adhere to this standard in future research. The shortcomings could be attended by coexistence with other logic languages.

A configurator approach using product requirement inputs given in natural language and configuration knowledge in OWL format is presented in [7]. The configuration process is divided into three steps. First, the customer requirements are mapped to product functions using function knowledge stored in a function ontology. Secondly, the product functions are mapped to product components (or modules) using product knowledge stored in a component ontology. Lastly, the obtained product configuration goes through an optimization process conducted using a Bayesian network.

OWL is also used to represent configuration knowledge used in a configurator in [8]. To extend the configuration knowledge of the OWL ontology, [8] also includes a set of rules in Semantic Web Rule Language (SWRL) to obtain a higher degree of expressiveness when modeling constraints. In implementation of their configurator, [8] use the JESS (Java Expert System Shell) library. The OWL axioms and SWRL rules of the configuration model is translated into facts and rules in JESS (Java Expert System Shell) which are then used in the configurator. An assessment of their approach is presented in [8] using a PC hardware configuration example and in [9] using a construction drilling machine as an example

A general domain ontology for assembly is proposed in [10]. The ontology includes sub-ontologies with knowledge on the three aspects of *product*, *process*, and *equipment* as originally proposed in [5]. The purpose of the modeling activities in [10] is to support reconfiguration activities; hence, the ontologies are specifically tailored for reconfigurable manufacturing equipment.

In [11] a framework for configuring robot cells in a "plug and produce" manner is presented. By use of an *interconnector module* the high-level task control is separated from the low-level device control. The interconnector module makes the task control independent of specific device syntax by providing an abstraction upon the device commands in the form of *skills*. Hereby, each device has a functional description in terms of skills and the behavior of each skill is represented by a state-chart. A process can be defined by a set of skills and a behavior sequence for each skill is defined by the state-charts. By matching the skills provided by the devices and the skills required by a given process, a preliminary set of devices can be chosen. Furthermore, based on the state-charts representing the behaviors of skills, discrete process planning can be performed.

An autonomous reconfiguration approach is presented by [12]. They demonstrate in a laboratory experiment how an ontology-based reconfiguration agent can autonomously reconfigure a modular, intelligent manufacturing system in response to changing requirements. The knowledge is modeled in OWL format, implemented using JENA, and reasoned on using the Pellet reasoner [13].

As indicated by the selected work presented above, much research has focused on using ontologies for modeling manufacturing knowledge and the use of ontologies for modeling the knowledge used in configuration. Software configurator tools have been successfully applied in many diverse markets for product configuration [14]. Despite this, the use of configurators for manufacturing systems is limited, with no present research on configurators for collaborative robots. Thus, there is a potential in research on how shop floor operators can benefit from a configurator in selecting a feasible hardware configuration.

3. Configuration Framework

The configuration framework consists of a graphical software configuration tool (from here on referred to as the configurator) and the supporting configuration knowledge referred to as the configuration model. The user interaction of the framework is done through the configurator where the user provides process and product information about the task. Based on these inputs, the configurator suggests suitable equipment modules using semantic knowledge stored in the configuration model. The offset for our configuration framework is within commercial off-the-shelf equipment. This also defines the granularity on which configuration is possible. To allow an easy and efficient exchange of hardware components, the commercial components must be adapted to standardized and "common" interfaces; both in terms of mechanic, energy, and control. Although a modular hardware architecture is considered a prerequisite for the configuration framework, the adaptation of commercial components into modules is out of scope for the work presented in this paper.

3.1. Configuration Model

The configuration model denotes the information relevant to the configuration task stored as semantic data. Based on the review of related work, see Sec. 2, we use an ontology-based knowledge modeling method in the configuration framework. Following the theory of [5,10], knowledge on manufacturing systems is covered by the three domains: Process, product, and equipment. We adopt this segmentation in our modeling of configuration knowledge resulting in three ontologies; one for each domain. Additionally, we have created two "common" ontologies, a *geometry ontology* and a *material ontology*. These ontologies represent knowledge applicable to a wide range of technical entities and systems, and they cover general concepts of their respective topics; e.g. 3D-shapes, 2D-shapes, and units of measure for the geometry ontology, and individual materials and their properties for the material ontology. The three domain ontologies will be elaborated in the following sections.

All the ontologies have been modeled in OWL DL using the graphical IDE Protégé 5 [15]. The Pellet reasoner [13] is used to infer additional knowledge (axioms) based on the asserted knowledge (axioms); e.g. if a gripper uses a pneumatic energy source, it is inferred to be a pneumatic gripper by the reasoner.

3.1.1. Product Ontology

The product ontology contains knowledge on products manipulated in the process by the robot. The high-level concept taxonomy of the product ontology is inspired from the structure proposed by [10]. The core concepts of the product ontology and the relations to the material and geometry ontologies are illustrated in Fig. 1.

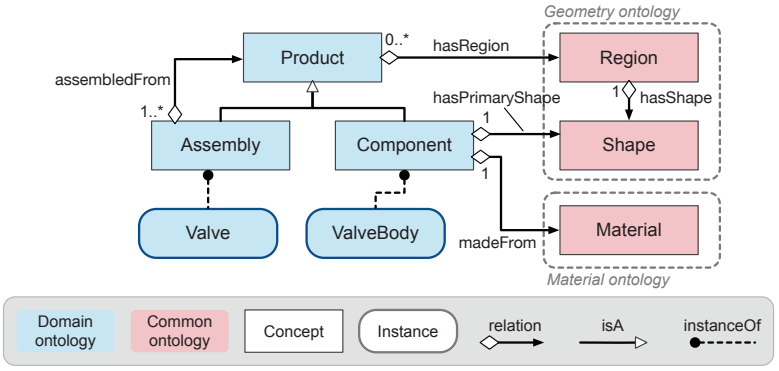

Fig. 1. The primary concepts and relations of the product ontology. Individuals are included to exemplify the instantiation of the concepts.

The top-most concept of the product ontology is the *product* which can either be a *component* or an *assembly*. We define a component as a single entity, not further decomposable into subcomponents. Hence, it cannot be a sub-assembly. The separation of assembly and component allows an explicit relation between a component and its material and geometric properties as exemplified in Fig. 1. A complex or 3rd party entity, like a bearing, can be modeled as a direct instance of *product*.

3.1.2. Process Ontology

The process ontology contains knowledge related to the task to be accomplished by the robot. Similar to the product ontology, the high-level concepts of the process ontology are inspired from [10]. Fig. 2. illustrates the top-most concepts of the process ontology and their relations. *Instances* are included in the figure to exemplify the extend of the concepts from which they are derived.

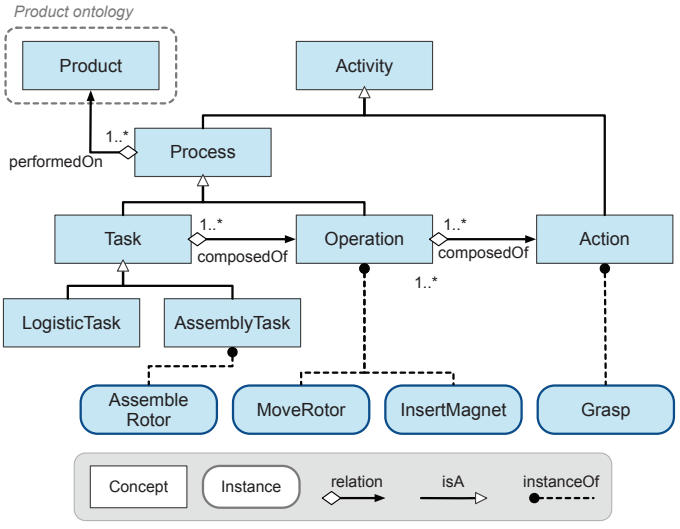

Fig. 2. The primary concepts and relations of the process ontology. Individuals are included to exemplify the instantiation of the concepts.

An *activity* is defined as any physical act, and the activity can be subsumed by either a *process* or an *action*. An action is an activity in which only the state of the actor is changed, and thus, actions naturally become low-level activities. A process is an activity performed on one or more products (component/assembly) resulting in a state change for the given products. The concept process is subsumed by either *task* or *operation*. A task is defined as a process performing a well-defined portion of work to achieve a production related goal. An operation is defined as a process changing the state of one or more products. Given that a task consists of one or more operations, the simplest task would be a single operation. Several generic operations are defined in the process ontology; e.g. *insert product into fixture* and *place product on surface*.

3.1.3. Equipment Ontology

The equipment ontology contains the knowledge on the equipment relevant to an industrial robotics system. This includes both conceptual knowledge on various equipment types and their relations, as well as knowledge on specific equipment instances. Thus, this ontology also contains the information on each of the specific commercial modules available to the robotic system.

Robotic systems are in industry used for a variety of different process domains; e.g. welding, machining, inspection and handling. We will in this work focus on the robot's ability to manipulate objects in assembly tasks. As such, we only model and illustrate the knowledge relevant to these tasks, but an equipment ontology focusing on another process domain would follow the same modeling principles. To further constrain the scope, we will in this paper focus on the manipulation-centric concepts of a robotic system. That is, the concepts that are part of the robot itself; typically including a robot arm, a robot tool, sensors, structural components, and control/computational components. The full equipment ontology cannot be visualized properly in this paper, but is made available for download[1]. The hierarchical structure of the primary concepts is shown in Fig. 3., and the primary relations between these concepts are shown in Fig. 4.

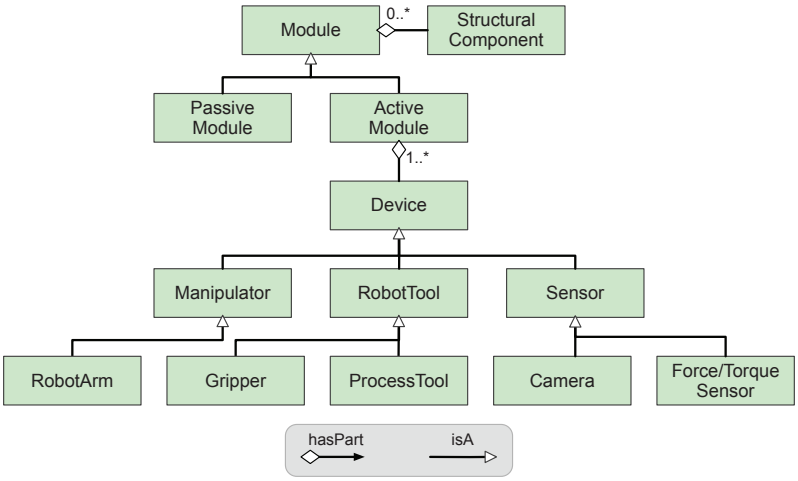

Fig. 3. The upper, central concepts of the equipment ontology.

[1] The most recent version of the equipment ontology can be downloaded from: http://tinyurl.com/csmanonto

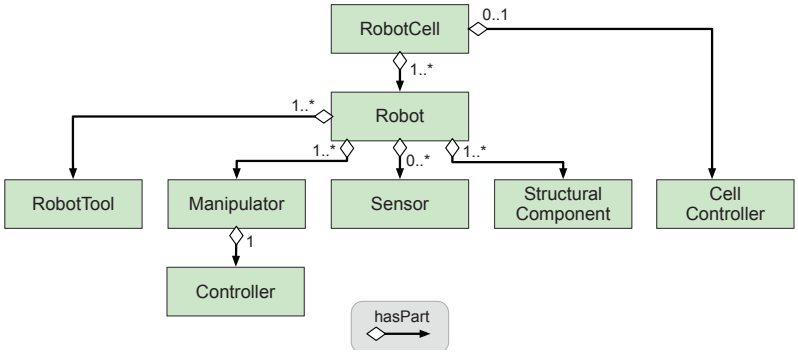

Fig. 4. The relations of the upper, central concepts of the equipment ontology.

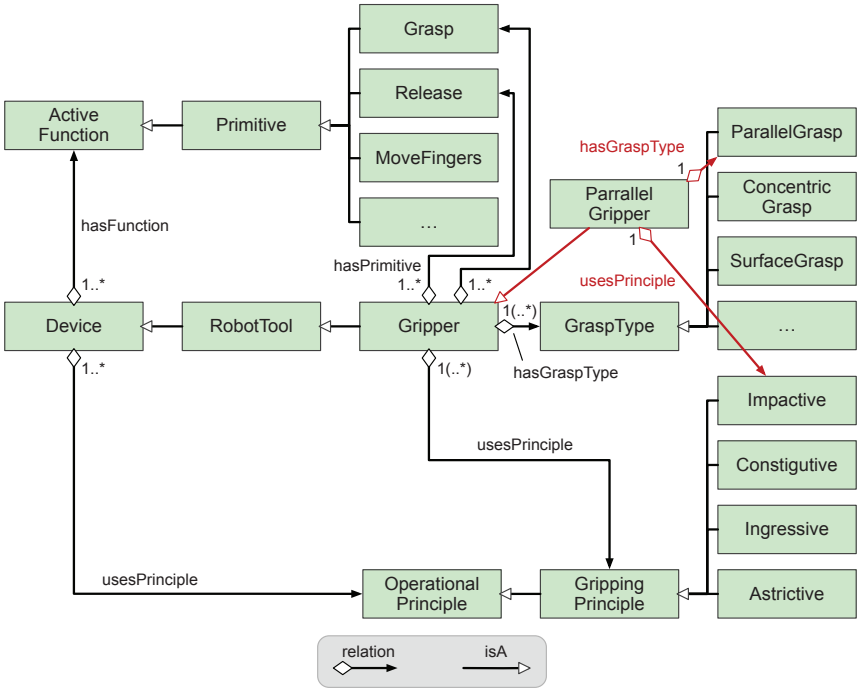

Fig. 5. Example of non-physical knowledge in the manipulation ontology. The figure illustrates some of the functional aspects related to a gripper. Further subsumption of the gripper concept is exemplified using the parallel gripper concept which further restricts the relations of the gripper concept. The classification of gripping principles is based on [17].

The concept of a *module* is subsumed by the *active module* which is a module containing at least one *device*. A device is defined as a controllable, mechatronic entity providing at least one *active function*, see Fig. 5. The equipment ontology furthermore covers non-physical concepts and relations such as functions and operating principles. Fig. 5. presents an example of non-physical knowledge in the equipment ontology. As shown in Fig. 5.Fig., a *gripper* is required to have at least one *grasp primitive* and at least one *release primitive*, but as given by the hierarchical structure a gripper is not restricted to only these two primitives. The subsumption of gripper by *parallel gripper* is shown to exemplify further restrictions on the relations.

The function concept represents abstract capabilities of *equipment*. The hierarchical structure of the function concept includes the concepts *task*, *skill*, and *primitive*, see Fig. 6. These concepts represent the software functions

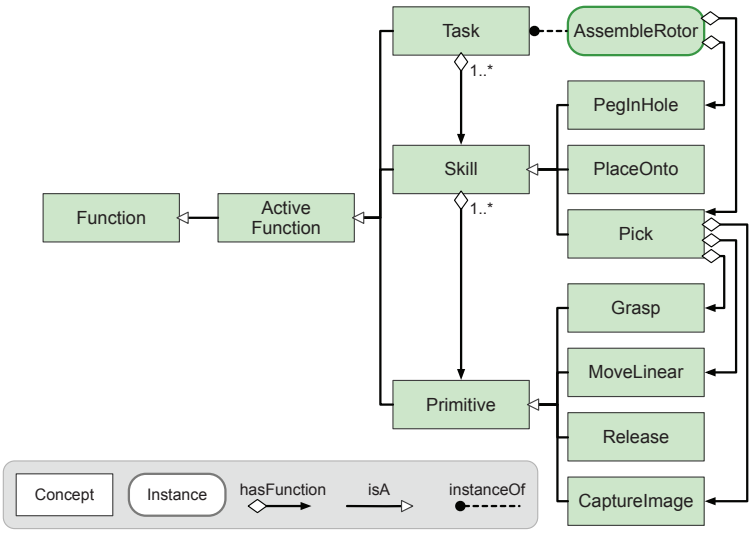

Fig. 6. The hierarchical structure of *function* concepts and their relations. The *function* concept is subsumed by the *task*, *skill*, and *primitive* used in skill-based programming.

used in the task programming following the skill-based programming paradigm described in [16]. The function concept and the subsumed concepts are modeled in the equipment ontology because they reflect equipment capabilities and not process requirements. However, the *task - skill - primitive* function structure is remarkable similar to the *task - operation - action* structure of the process ontology as both skills and operations are defined as effectuating a state-change to a given product. A key difference is though that the operations represent task-related goals from a human perspective, whereas the skills represent task-related goals from a robotic perspective. Consequently, the semantics of operations may not explicitly express all the activities needed to accomplish a task. That is, part of the activity in an operation can be implicit. For example, the operation *move product to bin* does not explicitly state that the product should be picked up first, but humans naturally infer this. This implicit knowledge is

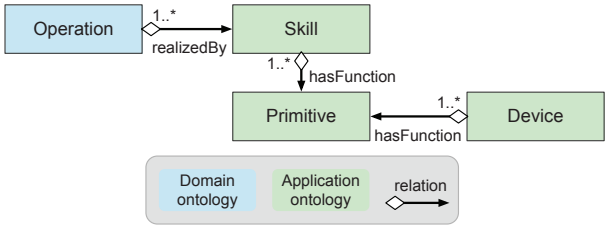

Fig. 7. Example of a relation between process and equipment knowledge. The link between operations and skills is modeled through the *realizedBy* relationship asserted in the equipment ontology.

needed in the robotic domain in order to successfully complete the task using skill-based programming. Mapping the process domain knowledge to the domain of equipment functions is done by asserting which skills are used to realize a given operation in the equipment ontology, see Fig. 7. Similarly, relations between product knowledge and equipment knowledge is also modeled in the equipment ontology. An example is shown in Fig. 8. where product shape is linked to the grasp type of a gripper.

3.2. Configurator

The *configurator* denotes the software tool using the configuration model to derive a feasible configuration based on user inputs. Thus, the configurator is designed to query the three domain ontologies along certain knowledge

propagations depending on the information provided by the user. Based on input of process and product information from the user, the configurator determines the required module types (e.g. *robot arm* or *camera*) and a set of feasible specific modules denoted *candidates* (e.g. *Schunk WSG50* or *ASUS Xtion*). The final selection within the set of candidates is done by the operator.

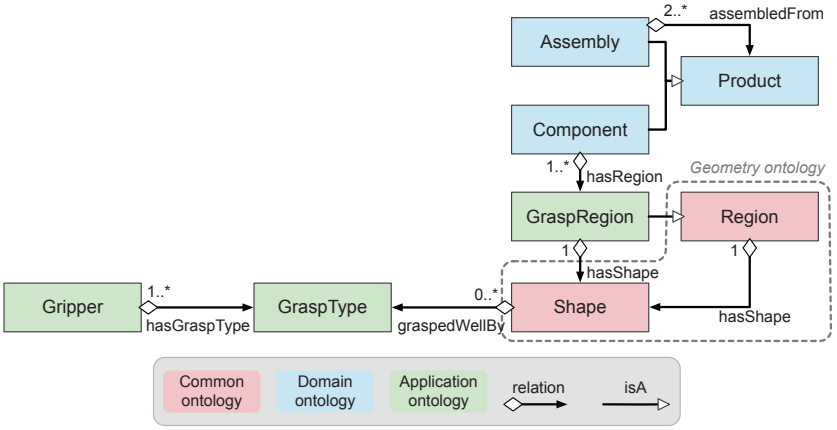

Fig. 8. Example of a relation between product and equipment knowledge. For each component one or more grasp regions are asserted in the product ontology. The link between the shape of a grasp region and a suitable grasp type is modeled through the *graspedWellBy* relationship which is asserted in the equipment ontology. Note that an assembly inherits the grasp regions of the components constituting the assembly.

At the back-end, the configurator contains mechanisms for querying the three ontologies and a set of instructions on how to connect the knowledge to derive the candidates. A few examples of such will be covered in this section. The configurator is implemented in C++ and the REDLAND and RASQAL libraries [18] provide the means to read,

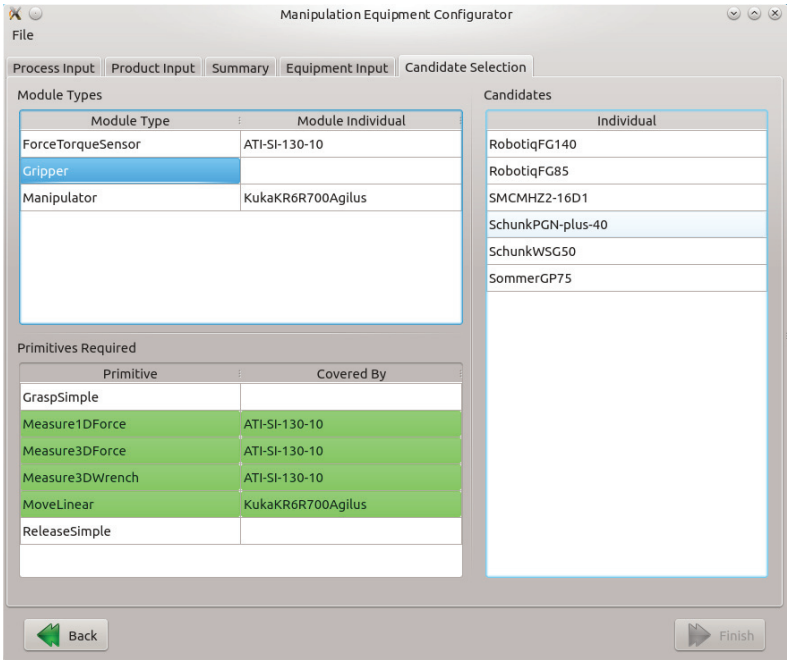

Fig. 9. Screenshot of the final candidate selection menu in the graphical user interface of the configurator.

write, and execute queries in the OWL files containing the ontologies.

At the front-end, the configurator provides the graphical user interface through which the operator interacts with the configurator. The user interface of the configurator is designed as a step-through "wizard". Fig. 9. depicts the final candidate selection menu of the user interface, and an overview of the configuration flow is illustrated in Fig. 10. The first step is input of process information. As the scope of the configurator is to determine a configuration for a single task, the user input for the process is given by sequencing operations, see Sec. 3.1.2. Using the relation

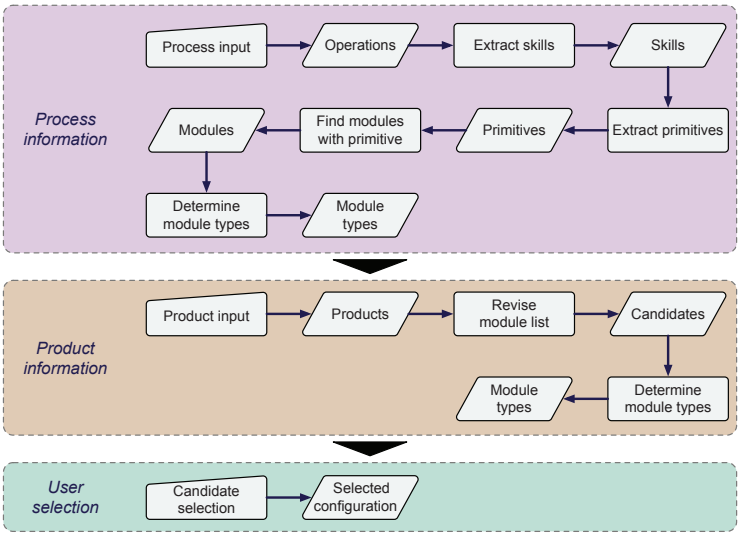

Fig. 10. Overview of the configuration flow in the configurator.

between operations and skills, the configurator maps the operations sequenced by the user into a set of required skills and from these into a set of required primitives. Based on the required primitives, the configurator extracts the individual modules providing these primitives, resulting in a list of all candidates providing at least one of the required primitives. The represented module types are determined by querying the module type for each individual candidate.

The next step is the input of product information where the user assigns a product to each operation in the task. The product knowledge including individual products is retrieved from the product ontology and displayed to the user. The configurator allows the user to create new products if needed which will then be inserted as axioms into the product ontology. The product information for each operation is then transferred to the skills realizing the operations, and the list of candidates already obtained from the process input is further refined. During this refinement, the candidates not suitable for the chosen products are removed from the list. In its current state, the configurator supports a maximum cardinality of one for each module type; i.e. maximum one robot arm, one gripper, one force/torque sensor, one camera etc. Thus, it does not consider the use of e.g. two grippers or the use of tool-changers.

After the input of process and product information, the configurator generates a summary of the inputs and determines the required skills, the required primitives, the required module types, and the possible candidates. The final step in the configuration process is the manual selection of specific modules from the list of candidates. As candidate list only contains valid modules and the selection is therefore a matter of user preference. The selection menu, see Fig. 9., lists the needed module types and the needed primitives. For each module type, the user is shown the candidates for that particular type. As modules are chosen, the covered primitives are marked green. The selection is finished once all required primitives have been covered by a specific module. The conclusion of the configuration process is the generation of a configuration file and a configuration report for use in the subsequent physical configuration of the robot.

4. Preliminary Feasibility Study

A preliminary, qualitative feasibility study of the configuration framework has been conducted. The purpose has been to examine the response of the configurator and determine if the candidate list, skill list, and primitive list indeed are qualified for a given task. The three industrial assembly tasks have been selected as a benchmark, see Fig.

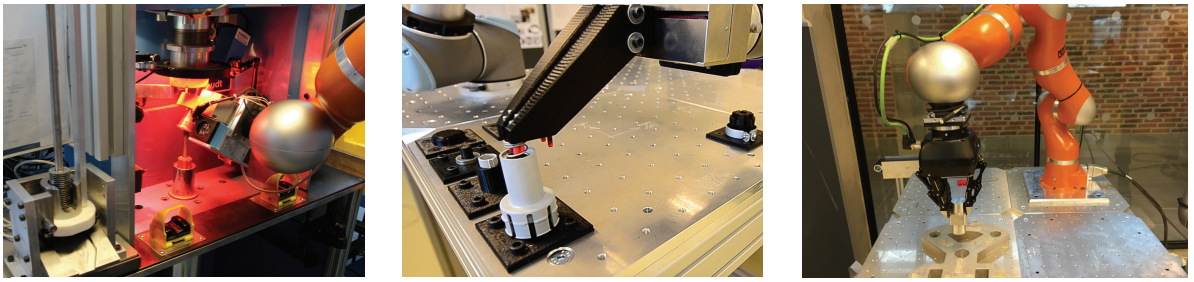

Fig. 11. Tasks used in the preliminary assessment of the configuration framework. Left: The assembly of a rotor from 10 metal components. Middle: The assembly of a spring-loaded plastic socket from four components. Right: Part assembly of the Cranfield benchmark [19].

11. For all three tasks, the hardware of the robot system has been set up by a robotics experts and subsequently the task has been programmed and executed using skill-based programming. The necessary operations, products, and equipment modules have been entered in the configuration model prior to the study. The equipment modules include six robot arms, nine grippers, and two force/torque sensors. The process and product information for each task has been entered in the configurator to obtain a list of suggested candidates.

The evaluation for each task has been: 1) the previously used configuration is present among the candidates, and 2) each possible configuration is deemed feasible in the given task by a robotics expert. For all three tasks, the configurator did suggest the modules previously used to solve the task. A review of all the proposed candidates for each task by an expert generally resulted in only viable configuration. However, one exception was noted. In the rotor assembly task, see Fig. 11., the force/torque measurements of the UR5 robot is not sufficiently accurate to robustly execute the task, however, this configuration (hence without an external F/T sensor) was possible. This indicates the need to model knowledge on sensor accuracy in the equipment ontology.

5. Conclusion

We have in this paper presented the knowledge modeling of both product, process and equipment knowledge to be used in configuration of a collaborative robot composed of commercial of the shelf components. We have described the design and implementation of a proof-of-concept configurator with a graphical user interface aiding the user in selecting a feasible configuration based on process and product input. To assess the feasibility of our approach, the configuration model obtained, and the configurator itself, a preliminary, qualitative feasibility study has been carried out. The results from the feasibility study indicate that the configurator does produce the expected output within the defined scope. An analysis of all the proposed candidates did however reveal some candidates which would not be physically suitable for the given task. These were included in the candidate list due to the scope set for the proof-of-concept implementation of the configurator and configuration model which does not yet assess all implications of the various process and product inputs. Further maturation and extension of both configurator scope and the underlying knowledge would significantly improve the obtained response from the configurator.
In conclusion, the approach used in the configuration framework is deemed feasible and further maturation and extension of the scope is possible following the same method as demonstrated in this work. In future work, we will assess the usability of the proposed configuration framework by conducting a number of user studies with shop floor operators from industry.

Acknowledgements

The financial support from Innovation Fund Denmark through the projects CARMEN and MADE is gratefully acknowledged.

References

[1] H.A. ElMaraghy, Flexible and reconfigurable manufacturing systems paradigms. International Journal of Flexible Manufacturing Systems, vol 17, pp. 261– 276, 2006.
[2] R. Hadar and A. Bilberg, Manufacturing Concepts of the Future – Upcoming Technologies Solving Upcoming Challenges, 2012, pp. 123–128.
[3] SPARC, Robotics 2020 multi-annual roadmap for robotics in Europe, 2015, Rev. B.
[4] C. Schou and O. Madsen, Towards shop floor hardware reconfiguration for industrial collaborative robots, 19th International Conference on Climbing and Walking Robots and Support Technologies for Mobile Machines (CLAWAR 2016), 2016, pp. 158-168.
[5] H.K Rampersad, Integrated and Simultaneous Design for Robotic Assembly (Product Development: Planning, Designing, Engineering), 1994.
[6] L. Ramos, Semantic web for manufacturing, trends and open issues: Toward a state of the art. Computers Industrial Engineering, 2015, vol. 90, pp. 444 – 460.
[7] F. Colace, M. De Santo, and P. Napoletano, Product configurator: an ontological approach, Ninth International Conference on Intelligent Systems Design and Applications, 2009, pp. 908–912.
[8] D. Yang, R. Miao, H. Wu, and Y. Zhou, Product configuration knowledge modeling using ontology web language, Expert Systems with Applications, vol. 36, 2009, pp. 4399 – 4411.
[9] D. Yang, M. Dong, and R. Miao, Development of a product configuration system with an ontology-based approach. Computer-Aided Design, vol. 40, 2008, pp. 863 – 878.
[10] N. Lohse, H. Hirani, and S. Ratchev, Equipment ontology for modular reconfigurable assembly systems, Int. J. of Flexible Manufacturing Systems, vol. 17, 2006, pp. 301–314.
[11] M. Naumann, K. Wegener, and R. Schraft, Control architecture for robot cells to enable plug'n'produce. International Conference on Robotics and Automation, 2007, pp. 287–292.
[12] Y. Alsafi and V. Vyatkin, Ontology-based reconfiguration agent for intelligent mechatronic systems in flexible manufacturing, Int. J. of Robotics and Computer-Integrated Manufacturing, vol. 26, 2010, pp. 381 – 391.
[13] E. Sirin, B. Parsia, B.C. Grau, A. Kalyanpur, and Y. Katz, Pellet: A practical owl-dl reasoner, Int. J. of Web Semantics: Science, Services and Agents on the World Wide Web, vol. 5, 2007, pp. 51 – 53.
[14] L. Hotz, A. Felfernig, M. Stumptner, A. Ryabokon, C. Bagley, and K. Wolter, Knowledge-based configuration: From research to business cases, chapter 6: Configuration knowledge representation and reasoning, Morgan Kaufmann Publishers, 2014.
[15] Stanford University, Protégé 5, 2015, Web page: http://protege.stanford.edu.
[16] C. Schou, J.S. Damgaard, S. Bøgh, and O. Madsen, Human-robot interface for instructing industrial tasks using kinesthetic teaching, 44th International Symposium on Robotics, 2013, pp. 1-6.
[17] G.J. Monkman, S. Hesse, R. Steinmann, and H. Schunk, Robot grippers, 2007, John Wiley & Sons.
[18] D. Beckett, REDLAND RDF Libraries, 2015, Web page: https://librdf.org. Version 1.0.17.
[19] K. Collins, A.J. Palmer, and K. Rathmill, Robot technology and applications, 1st robotics Europe conference, 1985, pp. 187–199.